



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/693,659	10/24/2003	Jeffrey P. Snover	MS1-1741US	9647
22801	7590	09/09/2009		
LEE & HAYES, PLLC 601 W. RIVERSIDE AVENUE SUITE 1400 SPOKANE, WA 99201			EXAMINER HICKS, MICHAEL J	
			ART UNIT	PAPER NUMBER
			2165	
			NOTIFICATION DATE	DELIVERY MODE
			09/09/2009	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

lhptoms@leehayes.com

Office Action Summary

Application No.

10/693,659

Applicant(s)

SNOVER ET AL.

Examiner

Michael J. Hicks

Art Unit

2165

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 19 June 2009.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-3, 5-11, 13-15, 17-21 and 23 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-3, 5-11, 13-15, 17-21, and 23 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
 2. ☐ Certified copies of the priority documents have been received in Application No. _____.
 3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) ☐ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO/SB08)
Paper No(s)/Mail Date _____
- 4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date _____
- 5) ☐ Notice of Informal Patent Application
- 6) ☐ Other: _____

DETAILED ACTION

1. Claims 1-3, 5-11, 13-15, 17-21, and 23 Pending.

Claims 4, 12, 16, and 22 Cancelled.

Continued Examination Under 37 CFR 1.114

2. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 6/19/2009 has been entered.

Response to Arguments

3. Applicant's arguments filed 6/19/2009 have been fully considered but they are not persuasive.

As per Applicants arguments that Murray in view of Young and further in view of Kriens fail to disclose the limitation of the at least one execution element comprising one selected from a group consisting of: a cmdlet, a function, a filter, and external script, and an external executable, Examiner respectfully disagrees. Examiner notes that one of the options in the list of selections is a function, and asserts that the term 'function' is broad

and that any predefined, operable piece of code which produces output and/or provides functionality could reasonably be construed to be a function. As such, the execution elements may be considered to comprise functions in that the execution element is associated with a known command and executed to enact that command.

As per Applicants arguments that Murray in view of Young and further in view of Kriens fail to disclose "resolving a parameter of each individual object-based command constituent to the sequence of object-based commands to a data type", Examiner respectfully disagrees. Examiner asserts that the citation of Kriens referred to in Applicants arguments (e.g. "if the Data Object were to hold an integer, then the meta-type is an instance of the IntegerType class. All accesses to the Data Object are forwarded to the meta-type") does disclose the limitation. Examiner notes that the limitation states that a parameter of each command is resolved to a datatype, which is interpreted by Examiner to mean that an datatype is identified for the parameter of the command to which the parameter of the command could be appropriately associated. The citation from Kriens clearly states that a parameter of the command (e.g. the type of data referred to by the Data Object) is used to map the command (e.g. Data Object) to meta-type (e.g. instance of a datatype, in this case the IntegerType class). Examiner believed this to be a correct reading of the claim language and the reference, however further clarification from Applicant is welcome if it is believed that Examiners reading of the intent of the limitation is in error.

As per Applicants arguments that Murray in view of Young and further in view of Kriens fail to disclose the limitations of "when data types of parameters of individual object-based commands are not natively supported by the operating environment, retrieving extended information comprising extended metadata and code, the extended metadata describing the data type and the code comprising additional instructions to populate an instance of the data type, that defines the data types and creating the instance of the data types for the parameter of each object-based command in the sequence that was resolved to one of the data types", and "'when data types of parameters of individual object-based commands are different than the data type specified by the property declared within the executable element class of the executable element associated with the individual object-based command, retrieving extended information comprising extended metadata and code", Examiner respectfully disagrees. Examiner notes that in Paragraph 13 of Applicants arguments, Applicant points out that Kriens teaches "'creat[ing] a new class for each possible user-defined type..., defin[ing] the types and generat[ing] the code to create instances of that type" (column 12 lines 56-60)". Examiner asserts that this citation indicates that non-predefined datatypes (e.g. non-native and non-matching datatypes), when encountered, are dealt with by creating an extended definition (e.g. descriptive metadata) for, and code to implement (e.g. create an instance of) the encountered datatype.

As per Applicants arguments that Murray in view of Young and further in view of Kriens fail to disclose the limitation of "the retrieving extended information comprises the

use of at least one of a property path mechanism, a key mechanism, a compare mechanism, a conversion mechanism, a globber mechanism, a relationship mechanism, and a property set mechanism", Examiner respectfully disagrees. Examiner notes, as discussed above, that the meta-types of Kriens are specialized instances of standard types (See, Column 15-Lines 14-20 as cited by Applicant in Applicants arguments, "each of [a] basic type [Strings, Numbers, Arrays, Structures, and Unions] can be represented with a meta-type class where each instance of this meta-type class further represents a user-defined data type. The meta-type class...defined for each basic type describes the behavior of data associated with a user-defined type"), and as such, Examiner asserts that the retrieving/generating of the extended information includes at least a relationship mechanism which relates the user defined data type to the basic type.

In view of the above arguments, the rejection of Claims 1-3, 5-11, 13-15, 17-21, and 23 will be updated to reflect amendments made to the claims and maintained.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

5. Claims 1-11, and 13-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Murray et al. (U.S. Pub. No. 2006/0235968 A1) in view of Young (U.S. Patent No. 6,782,531 B2) and further in view of Kriens (U.S. Patent No. 5,864,862)

As to claims 1, and 19, Murray et al. discloses a system that extends data types available to an operating environment, the system comprising:

a processor (See Murray et al. page 3, paragraph 0024); and

a memory, the memory being allocated for a plurality of computer-executable instructions which are loaded into the memory (See Murray et al. page 3, paragraph 0024) for execution by the processor, the computer-executable instructions comprising:

parsing a sequence of object-based commands into individual object-based commands (See Murray et al. page 7, paragraph 0090);

wherein the sequence of object-based commands comprise a command string and the individual object-based commands comprise respective constituent parts of the command string (See Murray et al. page 7, paragraph 0090 which clearly discloses that the commands may be CLI command sequences which are decomposed into an undetermined number of CLI actions.);

associating each individual object-based command with at least one execution element See Murray et al. page 6, paragraph 0085).

wherein the at least one execution element comprises one selected from a group consisting of: a cmdlet, a function, a filter, an external script and an external executable (See Murray et al. page 7, paragraph 0090 which clearly discloses that the one of the CLI actions may be a CLI read action, e.g. a read function.)

Murray et al. does not teach executing each execution element associated with each individual object-based command to produce output objects, wherein the execution of each execution element is execution dependent upon an execution-supporting operating environment.

Whereas, in the same field of endeavor, Young discloses a data processing system including a transaction processor pipeline made up of a number of pipeline stage (sub-component) and a parser for parsing data object; executing each execution element associated with each individual object-based command to produce output objects, wherein the execution of each execution element is execution dependent upon an execution-supporting operating environment (FIG. 2-3; col. 7, lines 15-61).

It would be obvious to one having ordinary skill in the art at the time the invention was made to modify Murray et al.'s invention with Young's invention to include a pipeline objects executed in different stage as sub-component for data processing. One would have been motivated to provide include such feature in order to provide a flexible, distributed system for performing calculations defined (Young's col. 8, lines 48-55).

The combination of Murray et al. and Young still does not teach resolving a parameter of each object-based command constituent to the sequence of object-based commands to a data type;

wherein a particular parameter of each individual object-based command is provided within an input object responsive to a property declared within a executable element class of the executable element associated with the individual object-based command comprising at least a name and data type;

when data types of parameters of individual object-based commands are not natively supported by the operating environment, retrieving extended information comprising extended metadata and code, the extended metadata describing the data type and the code comprising additional instructions to populate an instance of the data type, that defines the data types and creating the instance of the data types for the parameter of each object-based command in the sequence that was resolved to one of the data types;

when data types of parameters of individual object-based commands are different than the data type specified by the property declared within the executable element class of the executable element associated with the individual object-based command, retrieving extended information comprising extended metadata and code, the extended metadata describing the data type and the code comprising additional instructions to populate the instance of the data type, that defines the data types and creating the instance of the data types for the parameter of each object-based command in the sequence that was resolved to one of the data types;

and

wherein the retrieving extended information comprises the use of at least one of a property path mechanism, a key mechanism, a compare mechanism, a conversion mechanism, a globber mechanism, a relationship mechanism, and a property set mechanism.

Kriens teaches resolving a parameter of each object-based command constituent to the sequence of object-based commands to a data type (See Kriens column 15, lines 34-36, and see column 12, lines 56-60, and column 16, lines 11-19);

when data types of parameters of individual object-based commands are different than the data type specified by the property declared within the executable element class of the executable element associated with the individual object-based command, retrieving extended information comprising extended metadata and code, the extended metadata describing the data type and the code comprising additional instructions to populate the instance of the data type, that defines the data types and creating the instance of the data types for the parameter of each object-based command in the sequence that was resolved to one of the data types (See Kriens column 12, lines 56-60, column 15, lines 21-26, and column 15, lines 42-45, wherein "user defined types" are different than the data type specified by the property);

when data types of parameters of individual object-based commands are not natively supported by the operating environment, retrieving extended information comprising extended metadata and code, the extended metadata describing the data

type and the code comprising additional instructions to populate an instance of the data type, that defines the data types and creating the instance of the data types for the parameter of each object-based command in the sequence that was resolved to one of the data types (See Kriens column 12, lines 56-60, column 15, lines 21-26, and column 15, lines 42-45, wherein "user defined types" are non-native to operating environment)

and

wherein the retrieving extended information comprises the use of at least one of a property path mechanism, a key mechanism, a compare mechanism, a conversion mechanism, a globber mechanism, a relationship mechanism, and a property set mechanism (See Kriens column 15, lines 14-20).

It would be obvious to one having ordinary skill in the art at the time the invention was made to further modify Murray et al. as modified with the teachings of Kriens's invention to include resolving a parameter of each object-based command constituent to the sequence of object-based commands to a data type; wherein a particular parameter of each individual object-based command is provided within an input object responsive to a property declared within an executable element class of the executable element associated with the individual object-based command comprising at least a name and data type; when data types of parameters of individual object-based commands are not natively supported by the operating environment, retrieving extended information comprising extended metadata and code, the extended metadata describing the data type and the code comprising additional instructions to populate an instance of the data type, that defines the data types and creating the instance of the data types for the

parameter of each object-based command in the sequence that was resolved to one of the data types; when data types of parameters of individual object-based commands are different than the data type specified by the property declared within the executable element class of the executable element associated with the individual object-based command, retrieving extended information comprising extended metadata and code, the extended metadata describing the data type and the code comprising additional instructions to populate the instance of the data type, that defines the data types and creating the instance of the data types for the parameter of each object-based command in the sequence that was resolved to one of the data types; and wherein the retrieving extended information comprises the use of at least one of a property path mechanism, a key mechanism, a compare mechanism, a conversion mechanism, a globber mechanism, a relationship mechanism, and a property set mechanism

because it would introduce and accommodate further newly discovered data types relative efficiently to reduce the execution time.

As to claims 2, and 20, Murray et al. as modified discloses wherein the executing act comprises processing each execution element in order of each execution element's associated individual object-based commands in the sequence of object based commands (See Murray et al. page 6, paragraph 0087, and see Kriens column 20, lines 52-56).

As to claims 3, and 21, Murray et al. as modified discloses wherein the executing further comprises inputting into one or more execution elements output objects produced from one or more previously processed execution elements (See Murray et al. page 5, paragraph 0067, and see Murray et al. page 6, paragraph 0076, and see Young FIG. 2-3; Young col. 7, lines 15-61).

As to claim 5, Murray et al. as modified discloses comprising receiving the sequence of object-based commands via an object-based command pipeline (See Young column 7, lines 16-44, and see Young column 8, lines 1-10), wherein the sequence of object-based commands includes a wildcard and the processing further comprises producing a subset of the sequence of object-based commands based on the wildcard (See Murray et al. page 5, paragraph 0069, wherein it is inherent in any command language that character representing wildcard is accepted).

As to claim 6, Murray et al. as modified discloses further comprising receiving the sequence of object-based commands via an object-based command pipeline (See Young column 7, lines 16-44, and see Young column 8, lines 1-10), wherein the sequence of object-based commands includes a property set and the processing further comprises identifying a plurality of properties associated with the property set and processing the sequence of object-based commands based on the plurality of properties (See Murray et al. page 7, paragraph 0090).

As to claim 7, Murray et al. as modified discloses comprising receiving the sequence of object-based commands via an object-based command pipeline (See Young column 7, lines 16-44, and see Young column 8, lines 1-10), wherein the sequence of object-based commands includes a relation and the processing further comprises finding items that the sequence of object-based commands consume based on the relation (See Young column 7, lines 48-61, also see Murray et al. page 7, paragraph 0096, lines 16-20, wherein "sequential" is taught).

As to claim 8, Murray et al. as modified discloses further comprising receiving the sequence of object via an object-based command pipeline (See Young column 7, lines 16-44, and see Young column 8, lines 1-10), wherein the sequence of object-based commands comprises a property path, the property path comprises a series of components that provide navigation to a desired property of each object-based command in the sequence (See Murray et al. page 7, paragraph 0092, also see Murray et al. page 7, paragraph 0096, lines 16-20, wherein "sequential" is taught).

As to claims 9, and 23, Murray et al. as modified discloses wherein the sequence of object-based commands is associated with a first data type and the processing further comprising looking up a conversion for converting the first data type to the data type (See Murray et al. page 5, paragraph 0067, and see Murray et al. page 6, paragraph 0076).

As to claim 10, Murray et al. as modified discloses wherein each component comprises a property of each object-based command in the sequence, a method of each object-based command in the sequence, a field of each object-based command in the sequence, a third party property, or a third party object method (See Young column 8, lines 48-64).

As to claim 11, Murray et al. as modified discloses wherein the sequence of object-based is received as input to a subsequent command in the object-based command pipeline after processing the sequence of object-based commands (See Young column 7, lines 16-44, and see Young column 8, lines 1-10).

As to claim 13, Murray et al. as modified discloses wherein a component comprises a reference to registered code (See Murray et al. page 4, paragraph 0055).

As to claim 14, Murray et al. discloses a computer storage medium having computer executable instructions encoded thereon, which when executed by a computer perform operations facilitating resolution of partially unresolved input comprising:

receiving one or more parseable input objects (See Murray et al. page 7, paragraph 0090), the input objects being output from an already processed execution element that is associated with one or more object-based commands of a sequence of commands obtained within an execution-supporting operating environment, wherein the

execution of an execution element is execution dependent upon the execution-supporting operating environment in order to actually execute (See Murray et al. page 6, paragraphs 0086-0087);

wherein the sequence of commands comprise a command string and the one or more object-based commands comprise respective constituent parts of the command string (See Murray et al. page 7, paragraph 0090 which clearly discloses that the commands may be CLI command sequences which are decomposed into an undetermined number of CLI actions.);

retrieving extended information that defines the data type (See Murray et al. page 6, paragraph 0076); and

creating an instance of the data type (See Murray et al. page 5, paragraph 0069), wherein the receiving, retrieving, and creating acts facilitate resolution of partially unresolved input).

Murray et al. discloses the claimed invention but does not explicitly teach via an object-based command pipeline.

Whereas, in the same field of endeavor, Young discloses a data processing system including a transaction processor pipeline made up of a number of pipeline stage (sub-component) and a parser for parsing data object; and sending, with the method associated with a first pipeline sub-component, the at least one object to a next pipeline sub-component for processing, the next pipeline sub-component being one of the plurality of pipeline sub-components; and outputting information from at least one of

the plurality of pipeline sub-components, the information is the result of at least a portion of the processing performed by the object based pipeline (FIG. 2-3; col. 7, lines 15-61).

It would be obvious to one having ordinary skill in the art at the time the invention was made to modify Murray et al.'s invention with Young's invention to include a pipeline objects executed in different stage as sub-component for data processing. One would have been motivated to provide include such feature in order to provide a flexible, distributed system for performing calculations defined (Young's col. 8, lines 48-55).

The combination of Murray et al.'s invention with Young's still does not teach the one or more parseable input objects including content that uses a data type that is not natively supported by the execution-supporting operating environment or is different than a data type expected due to a property declared within an executable element class of one or more object-based commands comprising at least a name and datatype,

the extended information comprises extended metadata and code (See Kriens column 12, lines 56-60),

and wherein the extended metadata describes the data type and the code comprises additional instructions to populate an instance of the data type (See Kriens column 12, lines 56-60).

Kriens teaches the one or more parseable input objects including content that uses a data type that is not natively supported by the execution-supporting operating environment or is different than a data type expected due to a property declared within an executable element class of one or more object-based commands comprising at

least a name and datatype (See Kriens column 12, lines 56-60, column 15, lines 21-26, and see column 15, lines 42-45, wherein "user defined types" are non-native to operating environment or different than the data type expected);

the extended information comprises extended metadata and code,
and wherein the extended metadata describes the data type and the code comprises additional instructions to populate an instance of the data type.

It would be obvious to one having ordinary skill in the art at the time the invention was made to further modify Murray et al. as modified with the teachings of Kriens's invention to include the one or more parseable input objects including content that uses a data type that is not natively supported by the execution-supporting operating environment or is different than a data type expected due to a property declared within an executable element class of one or more object-based commands comprising at least a name and datatype, the extended information comprises extended metadata and code, and wherein the extended metadata describes the data type and the code comprises additional instructions to populate an instance of the data type.

because it would introduce and accommodate further newly discovered data types relative efficiently to reduce the execution time.

As to claim 15, Murray et al. as modified discloses wherein the one or more parseable input objects comprises a Windows Management Instrumentation (WMI) input, an ActiveX Data Object (ADO) input, an XML input, or a third party data format

(See Murray et al. page 6, paragraph 0077).

As to claim 15, Murray et al. as modified discloses wherein the one or more parseable input objects comprises a third party object that provides an additional property to an object supported natively within the execution-supporting operating environment (See Murray et al. page 5, paragraph 0064, and see Murray et al. page 6, paragraph 0079).

As to claim 18, Murray et al. as modified discloses wherein the one or more parseable input comprises an ontology service (See Murray et al. page 3, paragraph 0027, and see Murray et al. page 6, paragraph 0079, wherein "ontology" reads on "grammar").

Points of Contact

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Hicks whose telephone number is (571) 272-2670. The examiner can normally be reached on Monday - Friday 9:00a - 5:30p.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Neveen Abel-Jalil can be reached at (571)272-4074. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Michael J Hicks
Art Unit 2165
Phone: (571) 272-2670
Fax: (571) 273-2670

/Neveen Abel-Jalil/

Supervisory Patent Examiner, Art Unit 2165